# ASIPMEISTER -ADDING NEW RESOURCE – TUTORIAL

[ASIPMEISTER -ADDING NEW RESOURCE – TUTORIAL 1](#_Toc76452712)

[SET THE DIRECTORIES 2](#_Toc76452713)

[IMPLEMENT A NEW RESOURCE AND TEST 2](#_Toc76452714)

[minmax.vhd 2](#_Toc76452715)

[testbench.v 2](#_Toc76452716)

[INCORPORATE THE RESOURCE IN ASIPMEISTER 3](#_Toc76452717)

[minmax.fhm 4](#_Toc76452718)

[fhmdbstruct 25](#_Toc76452719)

[ADD CUSTOM INSTRUCTIONS 26](#_Toc76452720)

[USING EXISTING RESOURCES 26](#_Toc76452721)

[USING NEW RESOURCES 28](#_Toc76452722)

[USING INSTRUCTIONS 30](#_Toc76452723)

[ORIGINAL APPLICATION 30](#_Toc76452724)

[testMinMaxAPP/test.c 30](#_Toc76452725)

[APPLICATION WITH MMAXS (Existing Resources) 30](#_Toc76452726)

[testMinMaxSW/test.c 30](#_Toc76452727)

[APPLICATION WITH MMAX (New Resources) 30](#_Toc76452728)

[testMinMaxHW/test.c 30](#_Toc76452729)

[MODELSIM SIMULATIONS 30](#_Toc76452730)

## SET THE DIRECTORIES

1. Login to any ***i80labpcXX.ira.uka.de*** directly or using SSH or using X2Go Client. For example, login as ***asip-sajjad04*** into ***i80labpc02.ira.uka.de***
2. Open shell terminal from the start menu. It should be in your default home directory. Type “***pwd***”

sajjad@i80pc57:~:$pwd

/home/sajjad

1. Create a new directory for your Lab e.g. “***SS21”***

sajjad@i80pc57:~:$mkdir SS21

sajjad@i80pc57:~:$cd SS21/

1. Create a directory for your ASIP project in “***SS21”***

sajjad@i80pc57:~/SS21:$mkdir ASIPMeisterProjects

sajjad@i80pc57:~/SS21:$cd ASIPMeisterProjects/

1. Create a new directory for your lab session in “***ASIPMeisterProjects”*** e.g. “***8”***

sajjad@i80pc57:~/SS21/ ASIPMeisterProjects:$mkdir 8

sajjad@i80pc57:~/SS21/ ASIPMeisterProjects:$cd 8/

## IMPLEMENT A NEW RESOURCE AND TEST

1. Write a new VHDL file of the required resource, for example, **MINMAX** to calculate the minimum and maximum from given two numbers.

sajjad@i80pc57:~/SS21/ ASIPMeisterProjects/8:$vim minmax.vhd

### minmax.vhd

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.std\_logic\_arith.all;

entity minmax is

Port ( clock : in std\_logic; reset: in std\_logic; enb: in std\_logic;

din1 : in STD\_LOGIC\_VECTOR (31 downto 0);

din2 : in STD\_LOGIC\_VECTOR (31 downto 0);

doutMin : out STD\_LOGIC\_VECTOR (31 downto 0);

doutMax : out STD\_LOGIC\_VECTOR (31 downto 0)

);

end minmax;

architecture st of minmax is

begin

process (clock, reset, enb)

begin

if (signed(din1) < signed(din2)) then

doutMin <= din1;

else

doutMin <= din2;

end if;

if (signed(din1) > signed(din2)) then

doutMax <= din1;

else

doutMax <= din2;

end if;

end process;

end st;

1. Write a new VHDL or Verilog testbench file to test the new resource **MINMAX**.

sajjad@i80pc57:~/SS21/ ASIPMeisterProjects/8:$vim testbench.v

### testbench.v

`timescale 1ns / 1ps

module testb;

// Inputs

reg [31:0] din1;

reg [31:0] din2;

reg clock;

reg reset;

reg enb;

// Outputs

wire [31:0] doutMin;

wire [31:0] doutMax;

// Instantiate the Unit Under Test (UUT)

minmax uut (

.clock(clock),

.reset(reset),

.enb(enb),

.din1(din1),

.din2(din2),

.doutMin(doutMin),

.doutMax(doutMax)

);

initial begin

// Initialize Inputs

reset = 1;

clock = 1;

enb = 1;

din1 = 23;

din2 = 45;

// Wait 100 ns for global reset to finish

#1000;

// Initialize Inputs

din1 = 123;

din2 = 45;

// Add stimulus here

end

always

begin

clock = 1'b1;

#20; // high for 20 \* timescale = 20 ns

clock = 1'b0;

#20; // low for 20 \* timescale = 20 ns

end

endmodule

1. Create a new ModelSim project, add above files, compile them, and simulate the hardware to verify the functionality. Once the functionality is verified you can add it to the ASIPmeister via a FHM file.

/home/sajjad/SS21/ASIPMeisterProjects/8/ASIPmeister/share/fhmdb/workdb/FHM\_work

sajjad@i80pc57:~/SS21/ ASIPMeisterProjects/8:$cd ASIPmeister/share/fhmdb/workdb/FHM\_work

-bash-3.2$ cat minmax.fhm

## INCORPORATE THE RESOURCE IN ASIPMEISTER

1. Copy ASIPmeister Software to a new directory for example:

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8:$ cp -rf /AM/ASIPmeister .

1. Create FHM file of the resource from “minmax.vhd” using the steps in the Laboratory Script Section 4.4.
2. Copy this FHM file into “*ASIPmeister/share/fhmdb/workdb/FHM\_work*”

sajjad@i80pc57:~/SS21/ ASIPMeisterProjects/8:$cd ASIPmeister/share/fhmdb/workdb/FHM\_work

sajjad@i80pc57:~/SS21/ ASIPMeisterProjects/8/ ASIPmeister/share/fhmdb/workdb/FHM\_work:$vim minmax.fhm

### minmax.fhm

<?xml version="1.0" encoding="Shift\_JIS" ?>

<FHM>

<model\_name> minmax </model\_name>

<model>

<design\_level> behavior </design\_level>

<version> 1.0 </version>

<author> <![CDATA[ Joe Random Hacker ]]> </author>

<affiliation> <![CDATA[ Uni Karlsruhe ]]> </affiliation>

<model\_info> <![CDATA[ - ]]> </model\_info>

<parameter>

<parameter\_value key="bit\_width">

<value> 4 </value>

<value> 8 </value>

<value> 16 </value>

<value> 32 </value>

</parameter\_value>

</parameter>

<function\_description>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register function definition in behavior level

# parameter : bit\_width

if ($#ARGV != 0) {

print "number of parameters is wrong.\n";

print "usage : this\_script bit\_width\n";

exit (100);

}

$bit\_width = $ARGV[0];

$msb = $bit\_width - 1;

print <<FHM\_DL\_FOO;

/\*\* minmax \*/

function minmax {

input {

bit [31:0] din1;

bit [31:0] din2;

}

output {

bit [31:0] doutMin;

bit [31:0] doutMax;

}

control {

in clock;

in reset;

in enb;

}

protocol {

[enb == '1'] {

valid dout;

}

}

}

FHM\_DL\_FOO

exit (0);

]]>

</script>

</function\_description>

<function\_conv>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register function definition in behavior level

# parameter : bit\_width

if ($#ARGV != 0) {

print "number of parameters is wrong.\n";

print "usage : this\_script bit\_width\n";

exit (100);

}

$bit\_width = $ARGV[0];

$msb = $bit\_width - 1;

print <<FHM\_DL\_FOO;

/\*\* minmax \*/

function minmax {

input {

bit [31:0] din1;

bit [31:0] din2;

}

output {

bit [31:0] doutMin;

bit [31:0] doutMax;

}

control {

in bit clock;

in bit reset;

in bit enb;

}

protocol {

single\_cycle\_protocol {

enb = '1';

}

}

}

FHM\_DL\_FOO

exit (0);

]]>

</script>

</function\_conv>

<function\_port>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register port information in behavior level

# parameter : bit\_width

if ($#ARGV != 0) {

print "number of parameters is wrong.\n";

print "usage : this\_script bit\_width\n";

exit (100);

}

$bit\_width = $ARGV[0];

$msb = $bit\_width-1;

print <<FHM\_DL\_PORTS;

clock in bit ctrl

reset in bit ctrl

enb in bit ctrl

din1 in bit\_vector 31 0 data

din2 in bit\_vector 31 0 data

doutMin out bit\_vector 31 0 data

doutMax out bit\_vector 31 0 data

FHM\_DL\_PORTS

exit (0);

]]>

</script>

</function\_port>

<design>

<design\_lang> vhdl </design\_lang>

<instance>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register instance in behavior level

# parameter : instance\_name bit\_width

if ($#ARGV != 1) {

print "number of parameters is wrong.\n";

print "usage : this\_script instance\_name bit\_width\n";

exit (100);

}

$instance\_name = $ARGV[0];

$bit\_width = $ARGV[1];

$msb = $bit\_width - 1;

$signals = <<END\_SIGNALS;

-- Your signal declaration here

END\_SIGNALS

$vhdl = <<END\_VHDL;

-- Your vhdl code here

process (clock, reset, enb)

begin

if (signed(din1) < signed(din2)) then

doutMin <= din1;

else

doutMin <= din2;

end if;

if (signed(din1) > signed(din2)) then

doutMax <= din1;

else

doutMax <= din2;

end if;

end process;

END\_VHDL

{

print <<FHM\_DL\_COMMENTS;

FHM\_DL\_COMMENTS

}

print <<FHM\_DL\_TOP\_2;

-- int\_port : internal port

-- ext\_port : external port

-- Comment :

library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.std\_logic\_arith.all;

entity $instance\_name is

port (

FHM\_DL\_TOP\_2

print <<FHM\_DL\_PORTS;

clock : in std\_logic;

reset : in std\_logic;

enb : in std\_logic;

din1 : in STD\_LOGIC\_VECTOR (31 downto 0);

din2 : in STD\_LOGIC\_VECTOR (31 downto 0);

doutMin : out STD\_LOGIC\_VECTOR (31 downto 0);

doutMax : out STD\_LOGIC\_VECTOR (31 downto 0)

);

FHM\_DL\_PORTS

{

print <<FHM\_DL\_ARCH;

end $instance\_name;

architecture st of $instance\_name is

$signals

begin

$vhdl

end st;

FHM\_DL\_ARCH

}

exit (0);

]]>

</script>

</instance>

<entity>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register instance in behavior level

# parameter : instance\_name bit\_width

if ($#ARGV != 1) {

print "number of parameters is wrong.\n";

print "usage : this\_script instance\_name bit\_width\n";

exit (100);

}

$instance\_name = $ARGV[0];

$bit\_width = $ARGV[1];

$msb = $bit\_width - 1;

{

print <<FHM\_DL\_TOP;

entity $instance\_name is

port (

FHM\_DL\_TOP

}

print <<FHM\_DL\_PORTS;

clock : in std\_logic;

reset : in std\_logic;

enb : in std\_logic;

din1 : in STD\_LOGIC\_VECTOR (31 downto 0);

din2 : in STD\_LOGIC\_VECTOR (31 downto 0);

doutMin : out STD\_LOGIC\_VECTOR (31 downto 0);

doutMax : out STD\_LOGIC\_VECTOR (31 downto 0)

);

FHM\_DL\_PORTS

{

print <<FHM\_DL\_BOTTOM;

end $instance\_name;

FHM\_DL\_BOTTOM

}

exit (0);

]]>

</script>

</entity>

<testvector>

<testvector\_script>

<![CDATA[ ]]>

</testvector\_script>

</testvector>

<synthesis>

<parameter></parameter>

<synthesis\_script>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register synthesis script in behavior level

# parameter : instance\_name priority bit\_width

if ($#ARGV != 2) {

print "number of parameters is wrong.\n";

print "usage : this\_script instance\_name priority bit\_width\n";

exit (100);

}

$instance\_name = $ARGV[0];

$priority = $ARGV[1];

$bit\_width = $ARGV[2];

if ($priority eq "area"){

$priority\_const = "set\_max\_area 0";

}

elsif ($priority eq "performance"){

$priority\_const = "set\_max\_delay -from all\_inputs() -to all\_outputs() 0";

}

elsif ($priority eq "power"){

$priority\_const = "";

}

elsif ($priority eq "none"){

$priority\_const = "";

}

else {

print "priority $priority is not supported.\n";

exit(100);

}

{

print <<FHM\_DL\_END\_OF\_SCRIPT;

hdlin\_auto\_save\_templates = TRUE

analyze -f vhdl $instance\_name.vhd

elaborate $instance\_name

uniquify

$priority\_const

create\_clock -period 10 -waveform{0 5} clock

compile

write -hierarchy -output $instance\_name.db

report\_area

report\_timing

quit

FHM\_DL\_END\_OF\_SCRIPT

}

exit(0);

]]>

</script>

</synthesis\_script>

</synthesis>

</design>

<estimation>

<estimation\_data>

<library name="OSAKA">

<est\_type name="shape">

<est\_index name="area">

<unit> mm2 </unit>

<translate>

<translate\_value key="gate"> 4201.68 </translate\_value>

<translate\_value key="mm2"> 1 </translate\_value>

</translate>

<parameters name="">

<max>

<data bit\_width="4"> 0.1 </data>

<data bit\_width="8"> 0.1 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</max>

<min>

<data bit\_width="4"> 0.1 </data>

<data bit\_width="8"> 0.1 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

</typ>

</parameters>

</est\_index>

<est\_index name="aspect\_ratio">

<!-- Dummy yet -->

</est\_index>

<est\_index name="height">

<!-- Dummy yet -->

</est\_index>

<est\_index name="width">

<!-- Dummy yet -->

</est\_index>

</est\_type>

<est\_type name="timing">

<est\_index name="delay">

<unit> ns </unit>

<parameters name="">

<max>

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</max>

<min>

<data bit\_width="4"> 0.72 </data>

<data bit\_width="8"> 0.72 </data>

<data bit\_width="16"> 0.72 </data>

<data bit\_width="32"> 0.72 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 0.72 </data>

<data bit\_width="8"> 0.72 </data>

<data bit\_width="16"> 0.72 </data>

<data bit\_width="32"> 0.72 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</priority>

</typ>

</parameters>

</est\_index>

<est\_index name="delay\_fullpath">

<!-- Dummy yet -->

</est\_index>

</est\_type>

<est\_type name="power">

<est\_index name="static\_power">

<unit> mW </unit>

<parameters name="">

<max>

<data bit\_width="4"> 2.2203 </data>

<data bit\_width="8"> 4.4270 </data>

<data bit\_width="16"> 8.7214 </data>

<data bit\_width="32"> 17.2327 </data>

</max>

<min>

<data bit\_width="4"> 2.2153 </data>

<data bit\_width="8"> 4.3512 </data>

<data bit\_width="16"> 8.5400 </data>

<data bit\_width="32"> 17.0462 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 2.2159 </data>

<data bit\_width="8"> 4.4179 </data>

<data bit\_width="16"> 8.7033 </data>

<data bit\_width="32"> 17.2202 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 2.2203 </data>

<data bit\_width="8"> 4.4270 </data>

<data bit\_width="16"> 8.7214 </data>

<data bit\_width="32"> 17.2327 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 2.2153 </data>

<data bit\_width="8"> 4.3512 </data>

<data bit\_width="16"> 8.5400 </data>

<data bit\_width="32"> 17.0462 </data>

</priority>

</typ>

</parameters>

</est\_index>

</est\_type>

<est\_type name="function\_cycle">

<!-- Dummy yet -->

</est\_type>

<est\_type name="function\_power">

<!-- Dummy yet -->

</est\_type>

</library>

</estimation\_data>

<estimation\_method>

<est\_type name="shape">

<est\_index name="area">

<parameters name="">

<max>

<data bit\_width="4"> 0.1 </data>

<data bit\_width="8"> 0.1 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</max>

<min>

<data bit\_width="4"> 0.1 </data>

<data bit\_width="8"> 0.1 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

</typ>

</parameters>

</est\_index>

<est\_index name="aspect\_ratio">

<!-- Dummy yet -->

</est\_index>

<est\_index name="height">

<!-- Dummy yet -->

</est\_index>

<est\_index name="width">

<!-- Dummy yet -->

</est\_index>

</est\_type>

<est\_type name="timing">

<est\_index name="delay">

<parameters name="">

<max>

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</max>

<min>

<data bit\_width="4"> 0.72 </data>

<data bit\_width="8"> 0.72 </data>

<data bit\_width="16"> 0.72 </data>

<data bit\_width="32"> 0.72 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 0.72 </data>

<data bit\_width="8"> 0.72 </data>

<data bit\_width="16"> 0.72 </data>

<data bit\_width="32"> 0.72 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</priority>

</typ>

</parameters>

</est\_index>

<est\_index name="delay\_fullpath">

<!-- Dummy yet -->

</est\_index>

</est\_type>

<est\_type name="power">

<est\_index name="static\_power">

<parameters name="">

<max>

<data bit\_width="4"> 2.2203 </data>

<data bit\_width="8"> 4.4270 </data>

<data bit\_width="16"> 8.7214 </data>

<data bit\_width="32"> 17.2327 </data>

</max>

<min>

<data bit\_width="4"> 2.2153 </data>

<data bit\_width="8"> 4.3512 </data>

<data bit\_width="16"> 8.5400 </data>

<data bit\_width="32"> 17.0462 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 2.2159 </data>

<data bit\_width="8"> 4.4179 </data>

<data bit\_width="16"> 8.7033 </data>

<data bit\_width="32"> 17.2202 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 2.2203 </data>

<data bit\_width="8"> 4.4270 </data>

<data bit\_width="16"> 8.7214 </data>

<data bit\_width="32"> 17.2327 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 2.2153 </data>

<data bit\_width="8"> 4.3512 </data>

<data bit\_width="16"> 8.5400 </data>

<data bit\_width="32"> 17.0462 </data>

</priority>

</typ>

</parameters>

</est\_index>

</est\_type>

<est\_type name="function\_cycle">

</est\_type>

<est\_type name="function\_power">

</est\_type>

</estimation\_method>

</estimation>

</model>

<model>

<design\_level> synthesis </design\_level>

<version> 1.0 </version>

<author> <![CDATA[ Joe Random Hacker ]]> </author>

<affiliation> <![CDATA[ Uni Karlsruhe ]]> </affiliation>

<model\_info> <![CDATA[ - ]]> </model\_info>

<parameter>

<parameter\_value key="bit\_width">

<value> 4 </value>

<value> 8 </value>

<value> 16 </value>

<value> 32 </value>

</parameter\_value>

</parameter>

<function\_description>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register function definition in behavior level

# parameter : bit\_width

if ($#ARGV != 0) {

print "number of parameters is wrong.\n";

print "usage : this\_script bit\_width\n";

exit (100);

}

$bit\_width = $ARGV[0];

$msb = $bit\_width - 1;

print <<FHM\_DL\_FOO;

/\*\* minmax \*/

function minmax {

input {

bit [31:0] din1;

bit [31:0] din2;

}

output {

bit [31:0] doutMin;

bit [31:0] doutMax;

}

control {

in clock;

in reset;

in enb;

}

protocol {

[enb == '1'] {

valid dout;

}

}

}

FHM\_DL\_FOO

exit (0);

]]>

</script>

</function\_description>

<function\_conv>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register function definition in behavior level

# parameter : bit\_width

if ($#ARGV != 0) {

print "number of parameters is wrong.\n";

print "usage : this\_script bit\_width\n";

exit (100);

}

$bit\_width = $ARGV[0];

$msb = $bit\_width - 1;

print <<FHM\_DL\_FOO;

/\*\* minmax \*/

function minmax {

input {

bit [31:0] din1;

bit [31:0] din2;

}

output {

bit [31:0] doutMin;

bit [31:0] doutMax;

}

control {

in bit clock;

in bit reset;

in bit enb;

}

protocol {

single\_cycle\_protocol {

enb = '1';

}

}

}

FHM\_DL\_FOO

exit (0);

]]>

</script>

</function\_conv>

<function\_port>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register port information in behavior level

# parameter : bit\_width

if ($#ARGV != 0) {

print "number of parameters is wrong.\n";

print "usage : this\_script bit\_width\n";

exit (100);

}

$bit\_width = $ARGV[0];

$msb = $bit\_width-1;

print <<FHM\_DL\_PORTS;

clock in bit ctrl

reset in bit ctrl

enb in bit ctrl

din1 in bit\_vector 31 0 data

din2 in bit\_vector 31 0 data

doutMin out bit\_vector 31 0 data

doutMax out bit\_vector 31 0 data

FHM\_DL\_PORTS

exit (0);

]]>

</script>

</function\_port>

<design>

<design\_lang> vhdl </design\_lang>

<instance>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register instance in behavior level

# parameter : instance\_name bit\_width

if ($#ARGV != 1) {

print "number of parameters is wrong.\n";

print "usage : this\_script instance\_name bit\_width\n";

exit (100);

}

$instance\_name = $ARGV[0];

$bit\_width = $ARGV[1];

$msb = $bit\_width - 1;

$signals = <<END\_SIGNALS;

-- Your signal declaration here

END\_SIGNALS

$vhdl = <<END\_VHDL;

-- Your vhdl code here

process (clock, reset, enb)

begin

if (signed(din1) < signed(din2)) then

doutMin <= din1;

else

doutMin <= din2;

end if;

if (signed(din1) > signed(din2)) then

doutMax <= din1;

else

doutMax <= din2;

end if;

end process;

END\_VHDL

{

print <<FHM\_DL\_COMMENTS;

FHM\_DL\_COMMENTS

}

print <<FHM\_DL\_TOP\_2;

-- int\_port : internal port

-- ext\_port : external port

-- Comment :

library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.std\_logic\_arith.all;

entity $instance\_name is

port (

FHM\_DL\_TOP\_2

print <<FHM\_DL\_PORTS;

clock : in std\_logic;

reset : in std\_logic;

enb : in std\_logic;

din1 : in STD\_LOGIC\_VECTOR (31 downto 0);

din2 : in STD\_LOGIC\_VECTOR (31 downto 0);

doutMin : out STD\_LOGIC\_VECTOR (31 downto 0);

doutMax : out STD\_LOGIC\_VECTOR (31 downto 0)

);

FHM\_DL\_PORTS

{

print <<FHM\_DL\_ARCH;

end $instance\_name;

architecture st of $instance\_name is

$signals

begin

$vhdl

end st;

FHM\_DL\_ARCH

}

exit (0);

]]>

</script>

</instance>

<entity>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register instance in behavior level

# parameter : instance\_name bit\_width

if ($#ARGV != 1) {

print "number of parameters is wrong.\n";

print "usage : this\_script instance\_name bit\_width\n";

exit (100);

}

$instance\_name = $ARGV[0];

$bit\_width = $ARGV[1];

$msb = $bit\_width - 1;

{

print <<FHM\_DL\_TOP;

entity $instance\_name is

port (

FHM\_DL\_TOP

}

print <<FHM\_DL\_PORTS;

clock : in std\_logic;

reset : in std\_logic;

enb : in std\_logic;

din1 : in STD\_LOGIC\_VECTOR (31 downto 0);

din2 : in STD\_LOGIC\_VECTOR (31 downto 0);

doutMin : out STD\_LOGIC\_VECTOR (31 downto 0);

doutMax : out STD\_LOGIC\_VECTOR (31 downto 0)

);

FHM\_DL\_PORTS

{

print <<FHM\_DL\_BOTTOM;

end $instance\_name;

FHM\_DL\_BOTTOM

}

exit (0);

]]>

</script>

</entity>

<testvector>

<testvector\_script>

<![CDATA[ ]]>

</testvector\_script>

</testvector>

<synthesis>

<parameter></parameter>

<synthesis\_script>

<script>

<![CDATA[

#!/usr/bin/perl

# This script generates register synthesis script in behavior level

# parameter : instance\_name priority bit\_width

if ($#ARGV != 2) {

print "number of parameters is wrong.\n";

print "usage : this\_script instance\_name priority bit\_width\n";

exit (100);

}

$instance\_name = $ARGV[0];

$priority = $ARGV[1];

$bit\_width = $ARGV[2];

if ($priority eq "area"){

$priority\_const = "set\_max\_area 0";

}

elsif ($priority eq "performance"){

$priority\_const = "set\_max\_delay -from all\_inputs() -to all\_outputs() 0";

}

elsif ($priority eq "power"){

$priority\_const = "";

}

elsif ($priority eq "none"){

$priority\_const = "";

}

else {

print "priority $priority is not supported.\n";

exit(100);

}

{

print <<FHM\_DL\_END\_OF\_SCRIPT;

hdlin\_auto\_save\_templates = TRUE

analyze -f vhdl $instance\_name.vhd

elaborate $instance\_name

uniquify

$priority\_const

create\_clock -period 10 -waveform{0 5} clock

compile

write -hierarchy -output $instance\_name.db

report\_area

report\_timing

quit

FHM\_DL\_END\_OF\_SCRIPT

}

exit(0);

]]>

</script>

</synthesis\_script>

</synthesis>

</design>

<estimation>

<estimation\_data>

<library name="OSAKA">

<est\_type name="shape">

<est\_index name="area">

<unit> mm2 </unit>

<translate>

<translate\_value key="gate"> 4201.68 </translate\_value>

<translate\_value key="mm2"> 1 </translate\_value>

</translate>

<parameters name="">

<max>

<data bit\_width="4"> 0.1 </data>

<data bit\_width="8"> 0.1 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</max>

<min>

<data bit\_width="4"> 0.1 </data>

<data bit\_width="8"> 0.1 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

</typ>

</parameters>

</est\_index>

<est\_index name="aspect\_ratio">

<!-- Dummy yet -->

</est\_index>

<est\_index name="height">

<!-- Dummy yet -->

</est\_index>

<est\_index name="width">

<!-- Dummy yet -->

</est\_index>

</est\_type>

<est\_type name="timing">

<est\_index name="delay">

<unit> ns </unit>

<parameters name="">

<max>

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</max>

<min>

<data bit\_width="4"> 0.72 </data>

<data bit\_width="8"> 0.72 </data>

<data bit\_width="16"> 0.72 </data>

<data bit\_width="32"> 0.72 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 0.72 </data>

<data bit\_width="8"> 0.72 </data>

<data bit\_width="16"> 0.72 </data>

<data bit\_width="32"> 0.72 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</priority>

</typ>

</parameters>

</est\_index>

<est\_index name="delay\_fullpath">

<!-- Dummy yet -->

</est\_index>

</est\_type>

<est\_type name="power">

<est\_index name="static\_power">

<unit> mW </unit>

<parameters name="">

<max>

<data bit\_width="4"> 2.2203 </data>

<data bit\_width="8"> 4.4270 </data>

<data bit\_width="16"> 8.7214 </data>

<data bit\_width="32"> 17.2327 </data>

</max>

<min>

<data bit\_width="4"> 2.2153 </data>

<data bit\_width="8"> 4.3512 </data>

<data bit\_width="16"> 8.5400 </data>

<data bit\_width="32"> 17.0462 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 2.2159 </data>

<data bit\_width="8"> 4.4179 </data>

<data bit\_width="16"> 8.7033 </data>

<data bit\_width="32"> 17.2202 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 2.2203 </data>

<data bit\_width="8"> 4.4270 </data>

<data bit\_width="16"> 8.7214 </data>

<data bit\_width="32"> 17.2327 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 2.2153 </data>

<data bit\_width="8"> 4.3512 </data>

<data bit\_width="16"> 8.5400 </data>

<data bit\_width="32"> 17.0462 </data>

</priority>

</typ>

</parameters>

</est\_index>

</est\_type>

<est\_type name="function\_cycle">

<!-- Dummy yet -->

</est\_type>

<est\_type name="function\_power">

<!-- Dummy yet -->

</est\_type>

</library>

</estimation\_data>

<estimation\_method>

<est\_type name="shape">

<est\_index name="area">

<parameters name="">

<max>

<data bit\_width="4"> 0.1 </data>

<data bit\_width="8"> 0.1 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</max>

<min>

<data bit\_width="4"> 0.1 </data>

<data bit\_width="8"> 0.1 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 0.001 </data>

<data bit\_width="8"> 0.01 </data>

<data bit\_width="16"> 0.1 </data>

<data bit\_width="32"> 0.1 </data>

</priority>

</typ>

</parameters>

</est\_index>

<est\_index name="aspect\_ratio">

<!-- Dummy yet -->

</est\_index>

<est\_index name="height">

<!-- Dummy yet -->

</est\_index>

<est\_index name="width">

<!-- Dummy yet -->

</est\_index>

</est\_type>

<est\_type name="timing">

<est\_index name="delay">

<parameters name="">

<max>

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</max>

<min>

<data bit\_width="4"> 0.72 </data>

<data bit\_width="8"> 0.72 </data>

<data bit\_width="16"> 0.72 </data>

<data bit\_width="32"> 0.72 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 0.72 </data>

<data bit\_width="8"> 0.72 </data>

<data bit\_width="16"> 0.72 </data>

<data bit\_width="32"> 0.72 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 0.75 </data>

<data bit\_width="8"> 0.75 </data>

<data bit\_width="16"> 0.75 </data>

<data bit\_width="32"> 0.75 </data>

</priority>

</typ>

</parameters>

</est\_index>

<est\_index name="delay\_fullpath">

<!-- Dummy yet -->

</est\_index>

</est\_type>

<est\_type name="power">

<est\_index name="static\_power">

<parameters name="">

<max>

<data bit\_width="4"> 2.2203 </data>

<data bit\_width="8"> 4.4270 </data>

<data bit\_width="16"> 8.7214 </data>

<data bit\_width="32"> 17.2327 </data>

</max>

<min>

<data bit\_width="4"> 2.2153 </data>

<data bit\_width="8"> 4.3512 </data>

<data bit\_width="16"> 8.5400 </data>

<data bit\_width="32"> 17.0462 </data>

</min>

<typ>

<priority name="area">

<data bit\_width="4"> 2.2159 </data>

<data bit\_width="8"> 4.4179 </data>

<data bit\_width="16"> 8.7033 </data>

<data bit\_width="32"> 17.2202 </data>

</priority>

<priority name="delay">

<data bit\_width="4"> 2.2203 </data>

<data bit\_width="8"> 4.4270 </data>

<data bit\_width="16"> 8.7214 </data>

<data bit\_width="32"> 17.2327 </data>

</priority>

<priority name="power">

<data bit\_width="4"> 2.2153 </data>

<data bit\_width="8"> 4.3512 </data>

<data bit\_width="16"> 8.5400 </data>

<data bit\_width="32"> 17.0462 </data>

</priority>

</typ>

</parameters>

</est\_index>

</est\_type>

<est\_type name="function\_cycle">

</est\_type>

<est\_type name="function\_power">

</est\_type>

</estimation\_method>

</estimation>

</model>

</FHM>

1. Add the new hardware into the ASIPmeister resource list by editing “***ASIPmeister/share/fhmdb/******fhmdbstruct”*** and add the line “***<model>minmax</model>***” in the FHM\_WORK class.

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8/ASIPmeister/share/fhmdb/:$vim fhmdbstruct

### fhmdbstruct

<FHM\_Structure>

<library name="basicfhmdb">

<class name="computational">

<model>adder</model>

<model>adder0</model>

<model>alu</model>

<model>alu0</model>

<model>barrelshifter0</model>

<model>divider</model>

<model>divider0</model>

<model>extender</model>

<model>extender0</model>

<model>mini\_alu</model>

<model>multiplexor</model>

<model>multiplexor0</model>

<model>multiplier</model>

<model>multiplier0</model>

<model>rotator</model>

<model>shifter</model>

<model>shifter0</model>

</class>

<class name="storage">

<model>register</model>

<model>register0</model>

<model>registerfile</model>

<model>registerfile0</model>

</class>

</library>

<library name="workdb">

<class name="FHM\_work">

<model>browregfile</model>

<model>dmau0</model>

<model>dummy\_register</model>

<model>fwu</model>

<model>fwu0</model>

<model>genport0</model>

<model>imau0</model>

<model>mifu</model>

<model>pcu</model>

<model>pcu0</model>

<model>sleeper0</model>

<model>wire0</model>

<model>wire\_in</model>

<model>wire\_inout</model>

<model>wire\_out</model>

<model>clamp</model>

<model>stepsize</model>

<model>index</model>

<model>adpcm</model>

<model>adpcmdecode</model>

<model>adpcmdecode2</model>

<model>minmax</model>

</class>

</library>

</FHM\_Structure>

1. Set the ASIPmeister PATH to this ASIPmeister either in .bashrc or manually by PATH variable each time.

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8/ASIPmeister/share/fhmdb:$cd ../../../

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8:$

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8:$ PATH=/home/sajjad/SS21/ASIPMeisterProjects/8/ASIPmeister/bin/:$PATH

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8:$ which ASIPmeister

~/SS21/ASIPMeisterProjects/8/ASIPmeister/bin/ASIPmeister

## ADD CUSTOM INSTRUCTIONS

1. We will implement two minmax instruction using existing resources and using new resource.

### USING EXISTING RESOURCES

1. For each ASIPmeister CPU create a separate directory in “***ASIPMeisterProjects”***. For example, copy TEMPLATE PROJECT and rename it e.g., “***brownie”*** for ASIPmeister CPU “***browstd32.pdb***”

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8:$cp -r /home/asip00/epp/ASIPMeisterProjects/TEMPLATE\_PROJECT ./brownie

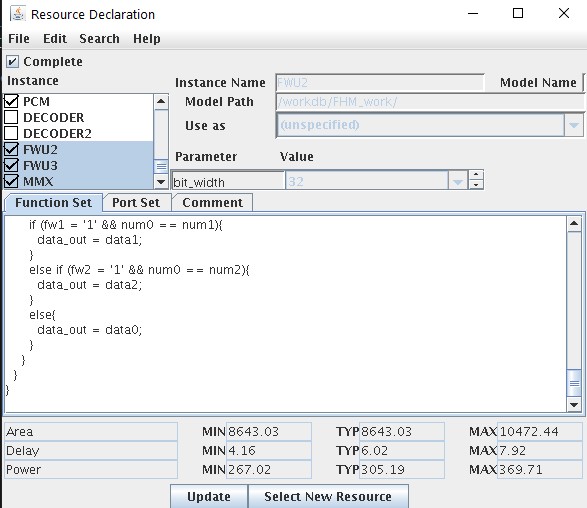
sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8:$ls

brownie

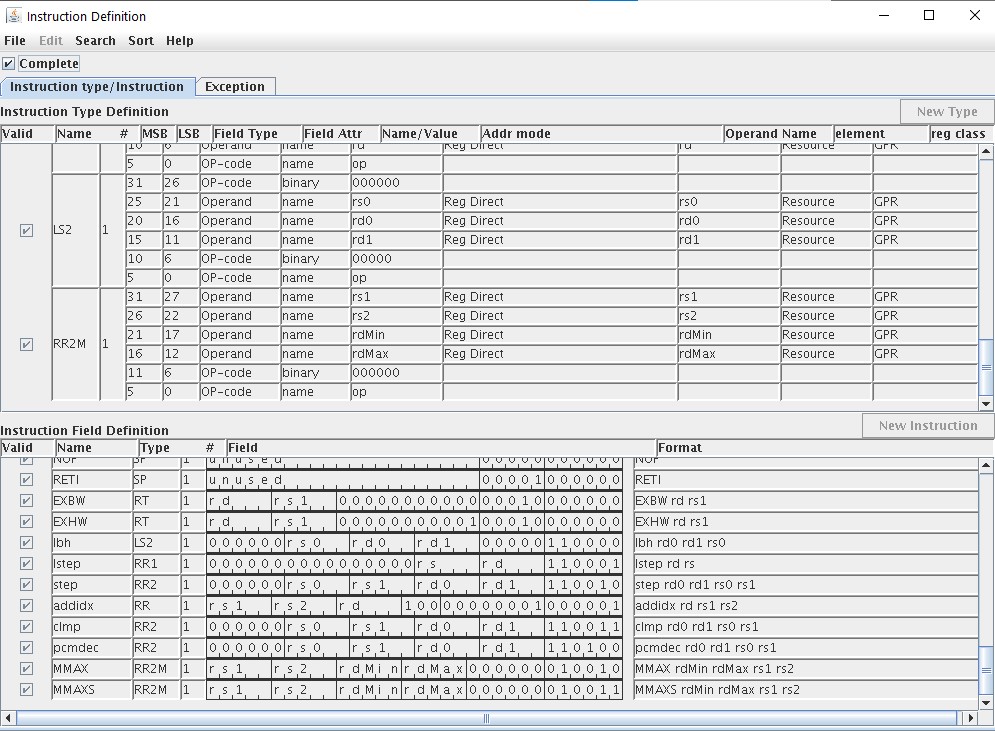
1. Set the parameters and settings of the ASIP project in “***env\_settings***”
2. Open ASIPMeister CPU in the respective directory i.e., in brownie

sajjad@i80pc57:~/SS21/Session1/ASIPMeisterProjects/brownie:$ASIPmeister browstd32.pdb &

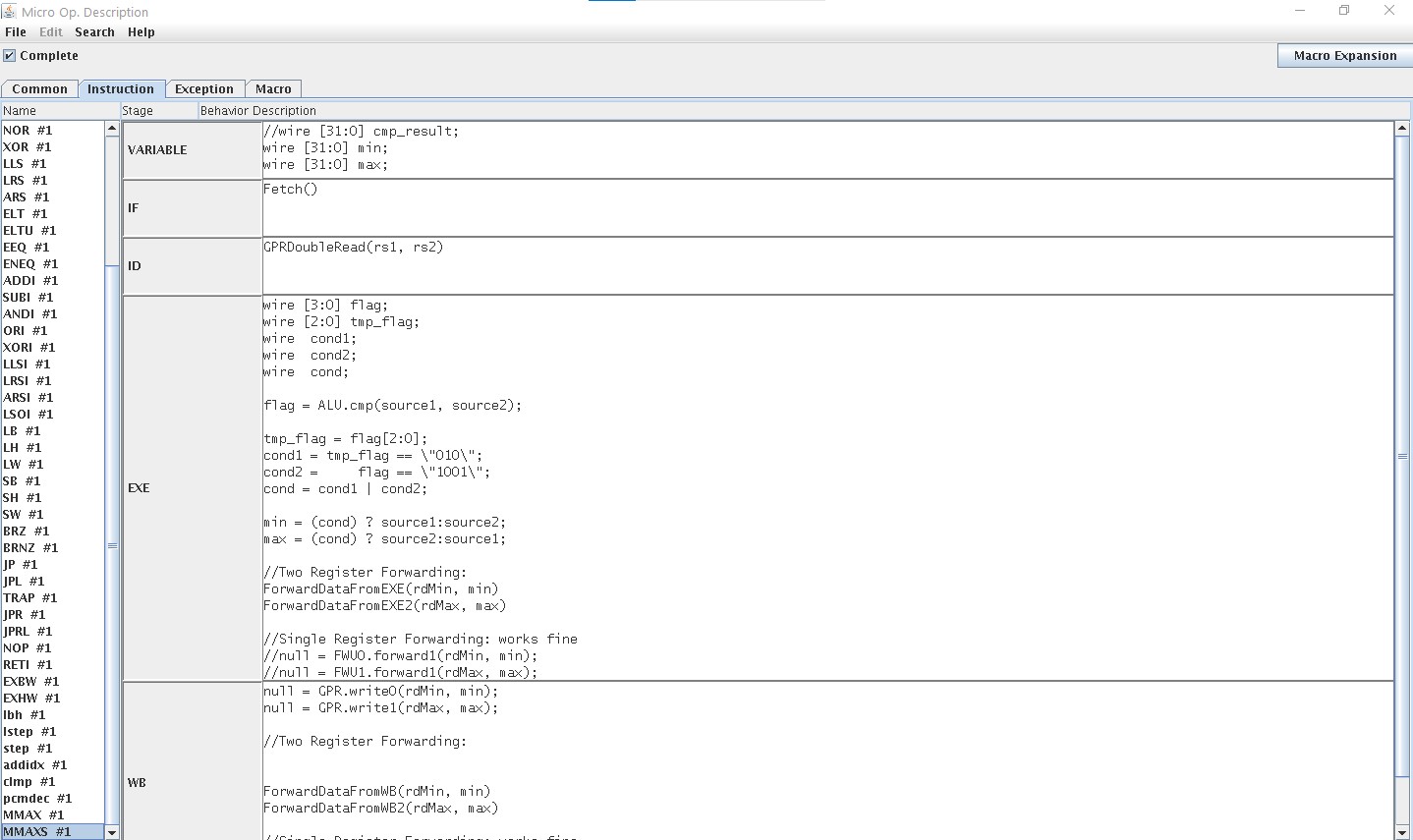
1. Modify the CPU in ASIPmeister. Resource Declaration will look like the following. “***MMX***” is the hardware instance of “***minmax.vhd***”. FWU2 and FWU3 are Forwarding Units for forwarding 2nd destination operand.



1. Then, create a new instruction formats with two input and two output registers. And create two different instructions which will use existing and new resources respectively.

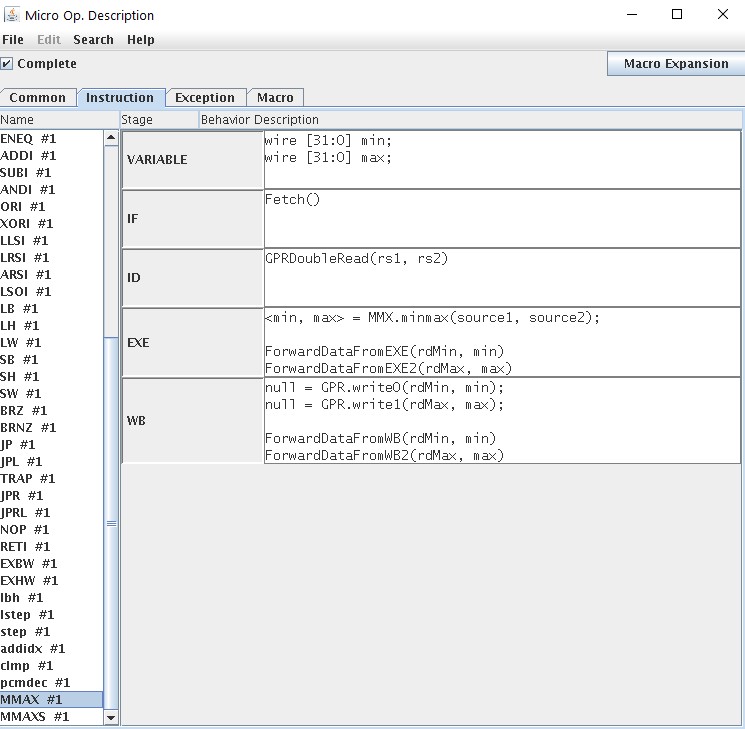


1. The Micro-Op description for the instruction MMAXS, which will be using the existing resources of ASIPmeister.

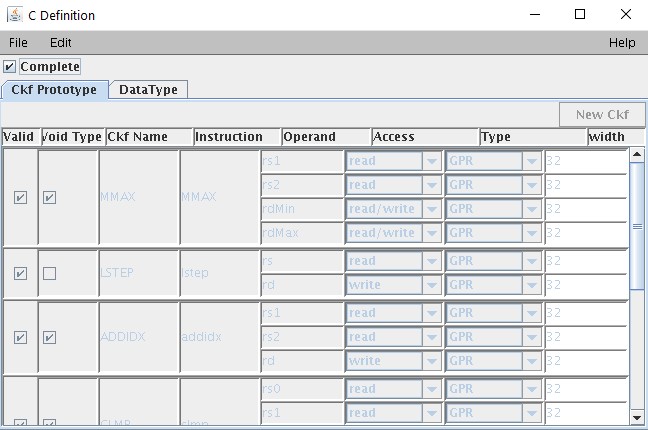


### USING NEW RESOURCES

1. The Micro-Op description for the instruction **MMAX**, which will be using the new resource **MMX** of ASIPmeister.



1. C Definition may look like this.



1. Generate the required compiler and VHDL files. A “***meister***” directory will be created in your ASIP project directory i.e. in “***brownie***”
2. Create a ModelSim project in the relevant directory of the current project, compile it and make it ready for the simulation.

## USING INSTRUCTIONS

1. Goto the “Applications” folder and create different applications to test each.

### ORIGINAL APPLICATION

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8/ brownie/Applications/ testMinMaxAPP:$ vim test.c

#### testMinMaxAPP/test.c

#define max(a, b) ( (a)>(b) ? (a):(b) )

#define min(a, b) ( (a)<(b) ? (a):(b))

unsigned int A[10] = { 32, 45, 0,0,0,0,0,0,0};

int main() {

A[4] = max(A[0], A[1]); // Maximum

A[5] = min(A[0], A[1]); // Minimum

return 0;

}

### APPLICATION WITH MMAXS (Existing Resources)

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8/ brownie/Applications/ testMinMaxSW:$ vim test.c

#### testMinMaxSW/test.c

unsigned int A[10] = { 32, 45, 0,0,0,0,0,0,0};

int main() {

\_\_asm\_\_ volatile (

"mmaxs %[out1], %[out2], %[op1], %[op2]\n\t"

: [out1] "=&r" (A[5]), [out2] "=&r" (A[4])

: [op1] "r" (A[0]), [op2] "r" (A[1])

);

return 0;

}

### APPLICATION WITH MMAX (New Resources)

sajjad@i80pc57:~/SS21/ASIPMeisterProjects/8/ brownie/Applications/ testMinMaxHW:$ vim test.c

#### testMinMaxHW/test.c

unsigned int A[10] = { 32, 45, 0,0,0,0,0,0,0};

int main() {

\_\_asm\_\_ volatile (

"mmax %[out1], %[out2], %[op1], %[op2]\n\t"

: [out1] "=&r" (A[5]), [out2] "=&r" (A[4])

: [op1] "r" (A[0]), [op2] "r" (A[1])

);

return 0;

}

### MODELSIM SIMULATIONS

1. ModelSim simulations produces following values:

testMinMaxAPP

-----------------

# \*\* Failure: SUCCESSFUL: Simulation End.

# Time: 2205 ns Iteration: 0 Process: /test/dmem File: /home/sajjad/SS21/ASIPMeisterProjects/8/dlx\_opt\_performance\_1/ModelSim/tb\_browstd32.vhd

# Break in Process dmem at /home/sajjad/SS21/ASIPMeisterProjects/8/dlx\_opt\_performance\_1/ModelSim/tb\_browstd32.vhd line 603

testMinMaxSW

-----------------

# \*\* Failure: SUCCESSFUL: Simulation End.

# Time: 1195 ns Iteration: 0 Process: /test/dmem File: /home/sajjad/SS21/ASIPMeisterProjects/8/dlx\_opt\_performance\_1/ModelSim/tb\_browstd32.vhd

# Break in Process dmem at /home/sajjad/SS21/ASIPMeisterProjects/8/dlx\_opt\_performance\_1/ModelSim/tb\_browstd32.vhd line 603

testMinMaxHW

-----------------

# \*\* Failure: SUCCESSFUL: Simulation End.

# Time: 1195 ns Iteration: 0 Process: /test/dmem File: /home/sajjad/SS21/ASIPMeisterProjects/8/dlx\_opt\_performance\_1/ModelSim/tb\_browstd32.vhd

# Break in Process dmem at /home/sajjad/SS21/ASIPMeisterProjects/8/dlx\_opt\_performance\_1/ModelSim/tb\_browstd32.vhd line 603